tions $\alpha = \frac{1}{2}$, $d_1 \geq d_0$ is equivalent to

$$k_1 f(f+1) \geq \frac{2(f-1)}{(\delta+1)} + C_1(f+1) \quad \text{where } \delta = (f-1)a,$$

$$k_1 = \left(1 + \delta + (f+1)\frac{\delta}{2}\right)^{-1},$$

$$C_1 = \left(1 + \delta + (g+1)\frac{\delta}{2}\right)^{-1}. \qquad (A.10)$$

Let $\phi = f - 1$, $\gamma = g - 1$. Then after tedious algebra, it can be shown that (A.10) is equivalent to the inequality

$$[-\phi^2 - 10\phi - \phi\gamma + 2\gamma]\,\phi^2 a^2 + [3\phi^2 + \gamma\phi^2 - 6\phi + \gamma\phi + 2\gamma]\,a + 2\phi^2 \geq 0. \qquad (A.11)$$

Recombining the terms of the left-hand side of (A.11), it becomes

$$(\gamma - \phi a - \gamma a)\,\phi^3 a + \phi^2 a[3\phi + \gamma - 10\phi a + 2\gamma a] + 2(\phi^2 - 2\phi^2 a + \gamma\phi a).$$

Since $\| h_0 \| = \| g_0 \|$ implies $\phi a = \gamma b$, the three terms of the above can be expressed as

$$\gamma - \phi\alpha - \gamma\alpha = \gamma(1-a) - \phi a \geq \gamma b - \phi b = 0,$$

$$3\phi + \gamma - 10\phi a + 2\gamma a \geq 2\phi(1 - b - a) + \phi(1 - 2a) + \gamma(1 - 2b) \geq 0,$$

$$\phi^2 - 3\phi^2 a + \gamma\phi a \geq \phi(1 - b - a) \geq 0.$$

The theorem is proven. The equality holds when 1) $\phi = \gamma = 0$, i.e., both X_0 and Y_0 are white noise, or 2) $a = b = \frac{1}{2}$.

Acknowledgment

The authors would like to express their sincere thanks to Dr. A. F. Petty, Dr. I. Jurkevich, and Dr. J. S. Lee for their help in data collection and computing, and for their stimulating discussions.

References

[1] L. Van Gool, P. Dewaele, and A. Oosterlinck, "Texture analysis anno 1983," *Comput. Vision, Graphics Image Processing*, vol. 29, pp. 336-357, 1985.

[2] A. Rosenfeld, "Survey, picture processing, 1985," *Comput. Vision, Graphics, Signal Processing*, vol. 34, pp. 204-251, 1985.

[3] J. T. Tou, "Pictural feature extraction and recognition via image modeling," in *Image Modeling*, A. Rosenfeld, Ed. 1981, pp. 391-421.

[4] A. K. Jain, "Partial differential equations and finite-difference models in imaging processing, Part 1: Image representation," *J. Optimiz. Theory Appl.*, vol. 23, pp. 65-91, 1977.

[5] ——, "Advances in mathematical models for image processing," *Proc. IEEE*, vol. 69, pp. 502-528, 1981.

[6] E. J. Delp, R. L. Kashyap, and O. R. Mitchell, "Image data compression using autoregression times series model," *Pattern Recognition Lett.*, vol. 11, pp. 313-323, 1979.

[7] R. L. Kashyap, "Characterization and estimation of two-dimensional ARMA models," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 736-745, 1984.

[8] R. L. Kashyap and A. Khotanzad, "A model-based method for rotation invariant texture classification," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 472-481, 1986.

[9] R. M. Haralick, "Statistical and structural approach to texture," *Proc. IEEE*, vol. 67, pp. 786-804, 1979.

[10] C. W. Therrien, "An estimation-theoretic approach to terrain image segmentation," *Comput. Vision, Graphics, Image Processing*, vol. 22, pp. 313-326, 1983.

[11] C. W. Therrien, T. F. Quatieri, and D. E. Dudgeon, "Statistical model-based algorithms for image analysis," *Proc. IEEE*, vol. 74, pp. 532-551, 1986.

[12] G. R. Cross and A. K. Jain, "Markov random field texture models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-5, pp. 25-39, 1983.

[13] R. Chellappa, and R. L. Kashyap, "Texture synthesis using 2-D noncausal autoregressive models," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 174-203, 1985.

[14] D. Tjøstheim, "Statistical spatial series modelling," *Adv. Appl. Prob.*, vol. 10, pp. 130-154, 1978.

[15] R. L. Kashyap, R. Chellappa, and A. Khotanzad, "Texture classification using features derived from random field models," *Pattern Recognition Lett.*, vol. 1, pp. 43-50, 1982.

[16] B. Julesz, "Cluster formation at various perceptual levels," *Methodol. Pattern Recognition*, pp. 297-315, 1968.

[17] B. Ashjari, "Computer detection and identification of a visually indiscernible texture mixture," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognition*, 1985, pp. 172-174.

[18] P. Diaconis and D. Freedman, "On the statistics of vision: The Julesz conjecture," *J. Math. Psychol.*, vol. 24, pp. 112-138, 1981.

[19] G. E. Keyte and J. T. Macklin, "SIR-B observations of ocean waves in the NE Atlantic," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-24, pp. 552-558, 1986.

[20] T. D. Allan, Ed., *Satellite Microwave Remote Sensing*. New York: Wiley, 1983.

[21] M. N. Youssef, "On the accuracy of forecasting telephone usage demand," *Bell Lab. Tech. J.*, vol. 63, pp. 819-849, 1984.

[22] T. Ozaki, "On the order of determination of ARIMA models," *Appl. Statist.*, vol. 26, pp. 290-301, 1977.

[23] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*, rev. ed. San Francisco: Holden-Day, 1976.

[24] F. M. Monaldo and D. R. Lyzenga, "On the estimation of wave slope- and height-variance spectra from SAR imagery," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-24, pp. 543-551, 1986.

[25] B. Kinsman, *Wind Wave*. Englewood Cliffs, NJ: Prentice-Hall, 1965.

[26] J. S. Lim and A. V. Oppenheim, "Enhancement and bandwidth compression of noisy speech," *Proc. IEEE*, vol. 67, pp. 1586-1640, 1979.

[27] H. Chang and J. K. Aggarwal, "Design of two-dimensional semicausal recursive filters," *IEEE Trans. Circuits Syst.*, vol. CAS-25, pp. 1051-1059, 1978.

[28] L. R. Rabiner and B. Gold, *Theory and Application of Digital Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975.

[29] J. S. Lee, "Speckle suppression and analysis for synthetic aperture radar images," *Opt. Eng.*, vol. 25, pp. 636-643, May 1986.

[30] W. A. Fuller, *Introduction to Statistical Time Series*. New York: Wiley, 1976.

[31] G. E. P. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive integrated moving average time series models," *J. Amer. Statist. Ass.*, vol. 65, pp. 1509-1526, 1970.

[32] A. I. Mcleod, "On the distribution of residual autocorrelations in Box-Jenkins models," *J. Roy. Statist. Soc. B*, vol. 40, pp. 296-302, 1978.

Two-Dimensional Linear Prediction Model-Based Decorrelation Method

ZHENYONG LIN AND Y. ATTIKIOUZEL

***Abstract*—This correspondence presents a unified feature extraction scheme, viz. two-dimensional (2-D) linear prediction model-based decorrelation method. By applying 2-D causal linear prediction model to decorrelate a textured image, the very heavy computation load required when using whitening operator to decorrelate the image, or the**

Manuscript received July 21, 1987; revised November 2, 1988. Recommended for acceptance by O. D. Faugeras. This work was supported by a research grant from the University of Western Australia.

The authors are with the Department of Electrical and Electronic Engineering, University of Western Australia, Nedlands WA 6009, Western Australia.

IEEE Log Number 8927501.

significant information loss when using gradient operator to approximately whiten the image is avoided. This texture model-based decorrelation method provides three sets of features to perform texture classification: the coefficients of the 2-D linear prediction, the moments of error residuals and the autocorrelation values. An optimum feature selection scheme using modified branch-and-bound method was introduced to reduce information redundancy. After feature selection, 100 percent classification accuracy was achieved for a 20 class texture problem. Experiments show that this feature extraction scheme is truly information lossless, effective, and fast.

Index Terms—**Feature extraction, image processing, texture analysis.**

I. Introduction

Texture is an important characteristic used in analyzing objects or regions of interest in an image. Texture features play an important role in image classification and analysis. In classification, texture features can be used to discriminate and label areas of an image, such as crop identification in an aerial photograph, and medical diagnosis of an X-ray photograph. Texture features can also be used in scene segmentation and identification in an image understanding or computer vision system, for example, in robot vision and industrial inspection. Therefore, the choice of texture features is the key in these applications.

Currently, there are a lot of approaches to choosing texture features. Some of these based on statistical techniques are: Fourier power spectrum [2], gray level run length [2], [6], Gray level difference [2], gray level co-occurrence matrices [1], [2], [7], [8], decorrelation method [3], and visually perceived properties [9]. Some of these based on parametric models are mosaic model [10], autoregressive model (AR) [11], [13], [22], and Markov random field (MRF) model [12], [14].

The motivation of this correspondence is to unify the parametric model-based approach with the statistical technique based approach for choosing texture features, viz. the 2-D linear prediction model-based decorrelation method. In the original decorrelation method of feature extraction [3], there are two major disadvantages: the computation of the autocorrelation shape features, or the use of a whitening operator to decorrelate the textured image, is computationally very expensive; some information will be lost in the whitening process, especially if one uses a gradient operator, such as the Laplacian or Sobel operator, to approximate the whitening operator. According to [14], the information loss is significant because the error residuals themselves contain only partial information of the original image. Due to these two reasons, it is not viable to use a decorrelation method in any practical situation.

A 2-D linear prediction model [15]–[17], [20] was widely used in many aspects of image analysis, for example, in image coding [17] and segmentation [15]. The two-dimensional autoregressive model, which is closely related to 2-D linear prediction model, was also used in texture classification [11], [13], [22]. Our objective in this paper is to use a 2-D causal linear prediction model to whiten a textured image to overcome the two drawbacks in the decorrelation method, thus making this feature extraction scheme much faster and information lossless. This can be achieved because the coefficients of the 2-D linear prediction model provide us with extra and very important information about texture, and its computation is fast. Note that the noncausal simultaneous autoregressive model (SAR) in [22] can also be used to whiten a textured image when using maximum likelihood estimation technique. However, the estimation of the 2-D linear prediction model involves only an inversion of a block Toeplitz autocorrelation matrix, and its computation is simpler and faster than the interative estimation scheme of the SAR model.

A Gaussian MRF model was also used for texture classification [14] and excellent results were obtained. Although the 2-D linear prediction model is a special case of the MRF model and SAR models under Gaussian assumption, for non-Gaussian textures, the probability structure of these models is different, and further problems may be posed to these two models for non-Gaussian textures or highly structured or macrotextures. Both [14] and [22] did not fully utilize information from the error field while the feature extracted from white noise field is a key feature as we will see from our experiment.

Our second modification to the original decorrelation method is to use autocorrelation values directly as features, as suggested in [14]. Because autocorrelation values themselves carry out enough texture information, it is not necessary to perform the time consuming autocorrelation shape feature calculation. This further simplifies the original method. With these two major modifications, our feature extraction scheme has proven a much faster and more powerful method. It produces a true information lossless feature set.

II. 2-D Linear Prediction Models

Two-dimensional linear prediction models have been widely reported in the literature [15], [17]. Consider $x(m, n)$ as a sample of a 2-D sequence of intensity image with m and n range over some finite lattices. The random field variable can then be represented by a 2-D prediction model [15], [17] as:

$$x_p(m, n) = \sum_{(k,j)\in\Pi}\sum_{k\ j} a(k, j) * x(m - k, n - j) + a_0 \quad (1)$$

where Π represents the nonzero support region or mask for prediction coefficients $a(k, j)$; a_0 represents a locally constant bias coefficient added to the input.

The prediction error is

$$\begin{aligned} e(m, n) &= x(m, n) - x_p(m, n) \\ &= x(m, n) - \sum_{(k,j)\in\Pi}\sum_{k\ j} a(k, j) * x(m - k, n - j) - a_0. \end{aligned} \quad (2)$$

Suppose $x(m, n)$ is an arbitrary stationary 2-D random field, then to minimize the variance of the prediction error,

$$E\left[\left(x(m, n) - x_p(m, n)\right)^2\right] \quad (3)$$

must be a minimum.

By varying the linear prediction coefficients $a(k, j)$, minimization of (3) gives a set of linear equations called the Normal equations:

$$\begin{aligned} C(p, q) &- \sum_{(k,j)\in\Pi}\sum_{k\ j} a(k, j) * C(p - k, q - j) \\ &- a_0 * \sum_{(m,n)\in W}\sum_{m\ n} x(m - p, n - q) = \beta^2 * \delta(p, q) \end{aligned} \quad (4)$$

where β represents the minimum variance of the error field of the 2-D linear prediction; $\delta(p, q)$ is the two-dimensional Kronecker delta function, W is an $M \times M$ estimation window; $C(p, q)$ represents the covariance coefficients of random field $\{x(m, n)\}$ which are estimated as:

$$C(k, j) = \sum_{(m,n)\in W}\sum_{m\ n} \left[x(m, n) * \left(x(m - k, n - j)\right)\right]. \quad (5)$$

Solving the Normal equations, the optimum coefficients of the 2-D linear prediction model can be obtained. A stochastic representation for a pixel $x(m, n)$ is

$$x(m, n) = x_p(m, n) + e(m, n). \quad (6)$$

For a causal mask, which is either a non symmetric half plane or quarter plane, the error will be white noise [22] if the prediction mask is made sufficiently large. The matrix of the normal equation is a block Toeplitz matrix which is always positive definite. There are many methods available for inversion of such matrices, such

as the Cholesky decomposition [20] or the more efficient method described in [19]. Usually in image processing applications, only a low order prediction model is required, this means that both the computation time of the covariance coefficients is less, and the calculations of the prediction coefficients and moments of error residuals are faster.

There are three methods for calculating the coefficients of the model according to the treatment of bias, and two techniques for estimating the covariance coefficients [17]. In this paper, the autocorrelation method was used to determine the range of summation in (5). It is assumed that the data are zero outside the $(M \times M)$ estimation window. The local mean linear prediction approach [17] was employed to calculate the coefficients of the 2-D linear prediction. The local mean of the data over the $(M \times M)$ estimation window is estimated, and is subtracted from every pixel within the $(M \times M)$ frame. The coefficients of the 2-D linear prediction model are then obtained by solving the normal equations (4) without bias.

III. Classification Scheme

In this application, a 3×3 quarter-plane prediction mask, i.e., an 8th order 2-D linear prediction model, was chosen. The 8 coefficients of the 2-D linear prediction and the moments of error residuals were chosen as features. The features that were derived from the error residuals are: variance β, skewness, and Kurtosis [3], [18]. As the coefficients of the 3×3 quarter-plane 2-D linear prediction were extracted from autocorrelation values $C(K, j)$, with $K = -2, -1, \cdots 2$, and $j = 0, 1, 2$, they cover most information contained in these values. There is no point in choosing these autocorrelation values as features. Therefore the additional autocorrelation values $C(K, j)$ $(K = -3, 3, j = 0, 1, 2; K = -3, -2, \cdots 3, j = 3)$ required for 4×4 quarter-plane 2-D linear prediction were chosen as features. Since $C(-3, 0) = C(3, 0)$, there were only 12 autocorrelation values. These values were normalized by dividing by $C(0, 0)$ as suggested in [14]. The above three sets of features give a feature vector with 23 components.

Having chosen features, we can adopt a Bayes decision rule or other standard rules for classification. A minimum distance classifier [14], [22] was chosen because of its simplicity and less computation requirements, which is especially important in the later feature selection stage. It can be described as

$$d(X^{(t)}, i) = \sum_{f=1}^{n} \left[X^{(t)}(f) - \mu^{(i)}(f)\right]^2 / \sigma^{2(i)}(f). \qquad (7)$$

First, the feature mean $\mu^{(i)}(f)$ and variance $\sigma^{(i)}(f)$ were calculated for each feature component f $(f = 1, 2, \cdots n$, n is feature dimension) of every class i $(i = 1, 2, \cdots K$, K is class number). A "leave-one-out" strategy was adopted, i.e., to take out a specific feature vector from a class as test sample $X^{(t)}$, the remaining feature vectors in that class and all the feature vectors in other classes were used to train the classifier, i.e., the mean $\mu(f)$ and variance $\sigma(f)$ were recomputed for that class using the remaining feature vectors from it. After calculating distance from $X^{(t)}$ to every class, the test sample $X^{(t)}$ was then assigned to a class i^* for which the distance in (7) is a minimum. This was repeated for every sample of each class.

IV. Experiment Results

We often find that many feature extraction schemes perform classification well, but with only a small group of textures. When texture classes are increased, especially when macrotextures are included, the classification accuracy drops quite a lot. Therefore, in our experiment, we chose 20 classes of textures from Brodatz [4], including both microtextures and macrotextures, to test robustness of our scheme. The number of these textures are: D68, D9, D84, D24, D19, D92, D12, D112, D29, D17, D4, D57, D77, D36, D38, D15, D2, D65, D3, D95. Each texture was digitized into a (512×512) image by a PC vision board installed in an IBM AT computer. These images were further reduced to a (256×256) resolution by averaging every four pixels. Each image was then subdivided into 16 subimages of (64×64) size with one column overlap. Thus each texture class had 16 samples.

TABLE I
Classification Results Using the 23-Dimensional Feature Vector for (64×64) Sized Subimage

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
6	0	0	0	0	0	14	0	0	0	0	0	0	0	0	2	0	0	0	0	0
7	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	14	0	0	0	0	0	0	1	0	0	0	0	0
9	0	1	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	1	0
13	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0
17	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16

Misclassification: 8 out of 320.

Note: Numbers 1–20 represent D68, D9, D84, D24, D19, D92, D12, D112, D29, D17, D4, D57, D77, D36, D38, D15, D2, D65, D3, D95 in Brodatz, respectively.

In order to make feature vector invariant to illumination changes, hence increasing classification accuracy, some commonly used preprocessing techniques were adopted. First, a histogram equalization over each subimage was performed. Then normalizing each subimage into a zero empirical mean and unit empirical variance was carried out. After preprocessing, the feature vectors mentioned in Section III were calculated. Classification experiment was conducted using these 23 features and minimum distance classifier. The results are shown in Table I. Classification accuracy is 97.5 percent.

V. Feature Selection

From experimental results we can see that our feature extraction scheme is powerful. But there still exist some problems: the information contained in the feature set is redundant, and some features in the set make classification accuracy worse, instead of making a contribution to it; the feature dimension is still too high. High dimension requires longer computation time and more memory space, it also makes the design of the classifier difficult if one wants to use an advanced classifier, such as the Bayes classifier in [22]. Furthermore it makes classification process slow, and not suitable for real time identification. Also, in some cases one might want classification accuracy as high as possible, for example, in medical diagnosis. For all above reasons, it is necessary to design an optimal feature selection scheme to reduce the feature dimension. Although the computation might be heavy in this process, because it is an off-line computation, once optimum feature set has been found, the calculation of this set and classification using this set will be much faster.

The computation of minimum distance classifier is simple and fast, we use therefore this classifier with "leave-one-out" strategy directly in our feature selection scheme to reduce the heavy computation load during feature selection process.

Define a binary-valued solution vector A as

$$A = (a_1, a_2, \cdots a_n)^t; \quad a_j = 0, 1; \quad j = 1, 2, \cdots n \tag{8}$$

where n is equal to the feature dimension and t represents the vector transposition.

The minimum distance classifier can be transformed into the following expression:

$$d(x^{(t)}, i) = \sum_{j=1}^{n} a_j \left[x^{(t)}(j) - \mu^{(i)}(j)\right]^2 / \sigma^{2(i)}(j) \tag{9}$$

where $a_j = 1$ means that the jth feature component has been selected; $a_j = 0$ means it has been excluded.

Define an assignment matrix AF, for test data $x^{(t)}$ of class k ($k = 1, 2, \cdots 20$). If the ith distance $d(x^{(t)}, i)$ is minimum, then the ith column in the kth row will be increased by one, i.e., assign class k to class i. The diagonal elements of AF will then represent the number correctly classified. The total classification accuracy will be the trace of AF, $TR(AF)$. The feature selection problem now becomes: to find the value of vector A to maximize $TR(AF)$. Obviously, this is a zero-one unconstrained nonlinear integer programming problem. A modified branch-and-bound algorithm was designed to solve this problem efficiently. The algorithm is as follows.

Step 1: For each of the present branches, set $a_j = 0$ for $j = 1, 2, \cdots n$, and j not being included in the branch record, calculate the corresponding $TR(AF)$ and store it in a matrix B which records both the branch which is under processing and the classification accuracy after the jth component of features is masked off in that branch, i.e., let $a_j = 0$. So each present branch has many subbranches.

Step 2: Find the maximum classification accuracy of all the subbranches just obtained and stored in matrix B.

Step 3: If the maximum classification accuracy of the subbranches is larger than or equal to that of the present branch then continue to Step 4, else stop.

Step 4: The maximum classification accuracy is used as the low bound. Any subbranch which has classification accuracy lower than the maximum will be discarded. The branch record will record only the subbranches which have maximum classification accuracy. Go to Step 1.

The schematic diagram of this algorithm is represented by the tree in Fig. 1.

Note that branches a_1, a_n, a_2, a_n, a_1, a_2 and a_1, a_2, a_n represent the same set of features. So at each branch level, those branches which represent the same set of features but with different feature sequence should be reduced to one branch so as to increase computation efficiency.

Comment: This algorithm is a tradeoff between branch-and-bound algorithm and sequential backward selection (SBS) algorithm [21]. It overcomes the large amount of computation still required in the branch-and-bound algorithm, thus making it computationally feasible, and it also partially eliminates the drawbacks in the SBS algorithm, i.e., once some feature components have been discarded, it does not allow any revision of their merit. In the presented algorithm, if a feature component is discarded in one branch, it can be assessed by other branches. Therefore the algorithm will get optimum or near-optimum feature sets.

Above feature selection scheme was applied to the 23 dimensional feature vectors calculated in Section IV. Nine optimum feature sets were obtained with feature dimension of 13, 12, 11, 10, 9, respectively, each with 100 percent classification accuracy.

During the feature selection process, the classification accuracy was nondecrease until reaching 100 percent. When proceeded further after reaching 100 percent accuracy with dimension of 9, the accuracy decreased monotonically. Hence, we got a group of key features (from most important to least important): $a[0, 1]$, Kurtosis, $C[2, 3]$, $C[-3, 2]$, $a[0, 2]$, $a[2, 0]$. Taking out any one of these will make significant decrease of classification accuracy.

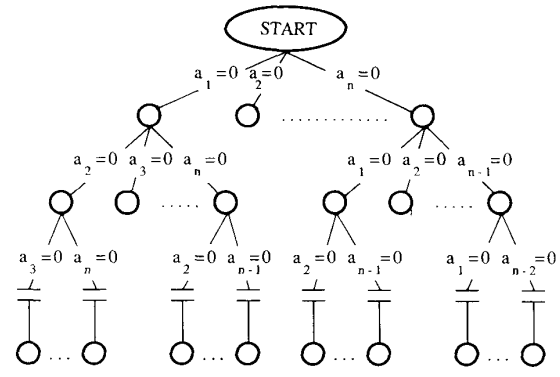

Fig. 1. Modified branch-and-bound.

In order to choose the best feature set from the nine groups, the same 20 classes of textures in Brodatz album were resampled with lower illumination condition and different area. The samples in the first group were used to train the minimum distance classifier with different feature set in the nine optimum groups. The samples in the second group were used as test samples. Two groups of the optimum feature set with dimension 13 misclassified only 2 samples of Field stone (D2) into Bubbles (D112). The remaining groups misclassified 3 samples of D2 into D112, all yielding classification accuracy of over 99 percent. This error is mainly due to the Field stone picture in Brodatz containing more dark holes, especially the area near the edge of the photograph, and lower illuminance condition. The best feature set, according to classification accuracy, is composed of $a[1, 0]$, $a[2, 0]$, $a[0, 1]$, $a[1, 1]$, $a[2, 1]$, $a[0, 2]$, β, Skewness, Kurtosis, $C(-3, 2)$, $C(-3, 3)$, $C(2, 3)$, $C(3, 0)$ or $C(3, 1)$.

VI. Summary and Conclusion

Experimental results show that this unified method is a true information lossless feature extraction scheme. It is more powerful than any single parametric model based approach or statistical technique based approach.

Because the computation of the coefficients of the 2-D linear prediction is fast, it is the cheapest white noise driven model, and therefore, makes this modified decorrelation method efficient, lossless and practical.

This scheme is suitable to a variety of textures, from microtexture to macrotextures. Therefore, it has a wide range of applications.

Experiments carried on an Intel 80386 CPU based computer Apricot VX show that the feature selection scheme given in this correspondence is suitable to feature dimension less than 30 with 20 classes. For higher feature dimension, one needs either to divide the feature vector into several groups or use a higher performance computer.

Acknowledgment

The authors wish to thank S. Chan and E. Lai for their help and F. Fung and K. K. Ly for their valuable suggestions and discussion.

References

[1] R. M. Haralick, "Statistical and structural approach to textures," *Proc. IEEE*, vol. 67, pp. 786-804, May 1979.

[2] J. Weszka, C. R. Dyer, and A. Rosenfeld, "A comparative study of texture measures for terrain classification," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, pp. 269-289, Apr. 1976.

[3] O. D. Faugeras and W. K. Pratt, "Decorrelation methods of texture feature extraction," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-2, pp. 323-332, July 1980.

[4] P. Brodatz, *Texture: A Photographic Album for Artists and Designers*. New York: Dover, 1956.

[5] C. H. Chen, "A study of texture classification using spectral features," in *Proc. IEEE 6th Int. Conf. Pattern Recognition*, Munich, Germany, Oct. 19-22, 1982, pp. 1074-1077.
[6] M. Galloway, "Texture analysis using gray-level-run lengths," *Comput. Graphics, Image Processing*, vol. 4, pp. 172-199, 1974.
[7] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, pp. 610-621, Nov. 1973.
[8] L. S. Davis, S. Johns, and J. K. Aggrawal, "Texture analysis using generalized co-occurrence matrices," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-1, pp. 251-259, July 1979.
[9] H. Tamara, S. Mori, and T. Yamawaki, "Texture features corresponding to visual perception," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, pp. 460-473, June 1978.
[10] N. Ahuja and A. Rosenfeld, "Mosaic models for texturers," *IEEE Trans. Pattern. Anal. Machine Intell.*, vol. PAMI-3, pp. 1-11, 1981.
[11] P. De Souza, "Texture recognition via autoregression," *Pattern Recognition*, vol. 15, pp. 471-475, 1982.
[12] G. R. Cross and A. K. Jain, "Markov random field texture models," *IEEE Trans. Pattern. Anal. Machine Intell.*, vol. PAMI-5, pp. 25-39, 1983.
[13] R. L. Kashyap and A. Khotanzad, "A model-based method for rotation invariant texture classification," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 472-481, July 1986.
[14] R. Chellappa and S. Chatterjee, "Classification of textures using Gaussian Markov random fields," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 959-963, 1985.
[15] C. W. Therrien, F. Outieri, and D. E. Dudgeon, "Statistical model-based algorithm for image analysis," *Proc. IEEE*, vol. 74, pp. 532-551, 1986.
[16] S. Ranganatn and A. K. Jain, "Two-dimensional linear prediction models—Part I: Spectral factorization and realization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 280-299, 1985.
[17] P. A. Maragos, R. W. Schafer, and R. M. Mersereau, "Two-dimensional linear prediction and its application to adaptive predictive coding of images," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 1213-1229, 1984.
[18] W. K. Pratt, *Digital Image Processing*. New York: Wiley, 1978.
[19] M. Wax and T. Kailath, "Efficient inversion of Toeplitz-block Toeplitz matrix," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, no. 5, 1983.
[20] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*. Englewood Cliffs, NJ: Prentice-Hall, 1978.
[21] *Handbook of Pattern Recognition and Image Processing*. New York: Academic, 1986.
[22] R. L. Kashyap, R. Chellappa, and A. Khotanzad, "Texture classification using features derived from random field models," *Pattern Recognition Lett.*, vol. 1, Oct. 1982.

Efficient Parallel Algorithms for Image Template Matching on Hypercube SIMD Machines

V. K. PRASANNA KUMAR AND VENKATESH KRISHNAN

***Abstract*—In this correspondence, we present efficient parallel algorithms for image template matching on hypercube SIMD machines of sizes N^2, $N^2 \times M^2$, and $N^2 \times K^2$ processing elements. For an $N \times N$ image and $M \times M$ window with N^2 PE's, we present a simple optimal parallel algorithm with $O(M^2 + \log N)$ time. The $N^2 \times M^2$ PE's case is solved in $O(\log N)$ time and the $N^2 \times K^2$ PE's case is solved in $O(M^2/K^2 + \log N)$ time. We also design a parallel algorithm for the N^2 PE's case using *constant* space/PE which runs in $O(M^2 * \log^* (M) + \log N)$ time. All these algorithms have superior time performance compared to known results.**

***Index Terms*—Hypercube, image processing, parallel algorithm, template matching.**

I. Introduction

Template matching is a basic operation in image processing and computer vision. It is used as a simple method for filtering, edge detection, image registration, and object detection. Template matching can be described as comparing a template (window) with all the possible windows of the image. Each position of the image will store the result of the window operation for which it is the top-left corner. Let $IMAGE(i, j)$ represent an $N \times N$ image where $i, j \in [0, N-1]$. Let $W(s, t)$ represent the template where $s, t \in [0, M-1]$. Then the result $C(i, j)$, $0 \le i, j \le N-1$ is as given below:

$$C(i,j) = \sum_{s=0}^{M-1} \sum_{t=0}^{M-1} IMAGE\big((i+s) \bmod N, (j+t) \cdot \bmod N\big) * W(s,t). \quad (1)$$

Note that the computation can be done in $O(N^2 \times M^2)$ time using a uniprocessor.

In this correspondence, we develop simple efficient parallel algorithms for template matching on hypercube arrays of various sizes. The results include image template matching on a hypercube with N^2 PE's in $O(M^2 + \log N)$ time. The storage space available in each PE is assumed to be $O(M)$. We also present an algorithm for a hypercube with $N^2 \times M^2$ PE's taking $O(\log N)$ time. This assumes constant storage space in each PE. The $N^2 \times K^2$ PE's algorithm is implemented in $O(M^2/K^2 + \log N)$ time. The storage in each PE is $O(M/K)$. We also show that these algorithms are optimal. Previous time bounds for the above three cases are $O(M * \max(M, \log N))$, $O(\log N * \log M)$, and $O(M^2/K^2 + (\log N * \log K))$, respectively [1], [2]. We also design a parallel algorithm for the N^2 PE case using *constant* space/PE which runs in $O(M^2 * \log^* (M) + \log N)$[1] time. This is an attractive algorithm if the window size is large. The main contribution of this paper is in designing simple elegant parallel algorithms for template matching. These results improve on the best known bounds in the literature [1], [2] and most of the solutions are optimal.

This correspondence is organized as follows. In the next section, we define the hypercube architecture and discuss some of its useful properties. We also define some data movement operations which will be used in our parallel algorithms. Section III deals with parallel algorithms for various cases and their optimality. Comparisons to known results are made in the last section.

II. A Hypercube SIMD Model

A hypercube SIMD computer is comprised of $N = 2^n$ processing elements (PE's), each having some local memory. The PE's are indexed 0 through $N-1$ and the pth PE is referred to as PE (p). All the PE's are synchronized and operate under the control of a single instruction stream. The PE's have enable/disable masks and a subset of PE's can be made to execute an instruction. The set of enabled PE's can be changed from instruction to instruction. Each PE has a local memory of size l, an ALU, a memory address register, temporary registers, a mask bit register, and an index register. The index register of PE (p) stores the index p of the PE.

The PE's are interconnected to provide inter-PE communica-

Manuscript received October 10, 1986; revised November 3, 1988. This work was supported in part by NSF Grant IRI-8710836 and by grants from the U.S.C. Faculty Research and Innovation Fund and the Powell Foundation.

The authors are with the Department of Electrical Engineering—Systems, University of Southern California, Los Angeles, CA 90089.

IEEE Log Number 8926699.

[1] $\log^* M = \min\{K \mid \log \log \log \cdots K \text{ times } (M) \le 1\}$.